Designing rational filter functions for solving eigenvalue problems by contour integration

Marc Van Barel*

May 15, 2015

Abstract

Solving (nonlinear) eigenvalue problems by contour integration, requires an effective discretization for the corresponding contour integrals. In this paper it is shown that good rational filter functions can be computed using (nonlinear least squares) optimization techniques as opposed to designing those functions based on a thorough understanding of complex analysis. The conditions that such an effective filter function should satisfy, are derived and translated in a nonlinear least squares optimization problem solved by optimization algorithms from Tensorlab. Numerical experiments illustrate the validity of this approach.

keywords: linear, polynomial, nonlinear eigenvalue problems; contour integration; filter function; rational approximation; nonlinear least squares; resolvent

MSC: Primary: 65-F15, Secondary: 47-J10, 30-E05.

1 Introduction

In this paper, the following *eigenvalue problem* is considered. Given an integer $m \geq 1$, a (bounded) domain $\Omega \subset \mathbb{C}$ and a matrix-valued function $T : \Omega \to \mathbb{C}^{m \times m}$ analytic in Ω, we want to compute the values $\lambda \in \Omega$ (eigenvalues) and $v \in \mathbb{C}^m$, $v \neq 0$ (eigenvectors) such that

$$T(\lambda)v = 0.$$

*KU Leuven, Department of Computer Science, Celestijnenlaan 200A, B-3001 Leuven (Heverlee), Belgium. marc.vanbarel@cs.kuleuven.be. The research was partially supported by the Research Council KU Leuven, project OT/10/038 (Multi-parameter model order reduction and its applications), PF/10/002 Optimization in Engineering Centre (OPTEC), by the Fund for Scientific Research–Flanders (Belgium), G.0828.14N (Multivariate polynomial and rational interpolation and approximation), and by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization). The scientific responsibility rests with its author(s).

Note that this formulation reduces to the linear eigenvalue problem in case $T(z) = A - zB$, and to the polynomial eigenvalue problem when $T(z)$ is a polynomial matrix. If the problem size m is equal to 1, then the problem reduces to that of computing all the zeros λ of the analytic scalar function T in the domain Ω.

The number of eigenvalues could be large, e.g., when m is large, or in case of a polynomial eigenvalue problem when the degree of the polynomial matrix is large. In several applications, one is not interested in *all* eigenvalues but only in those lying in a certain region(s) of the complex plane. Therefore, we can reduce the original problem of finding *all* eigenvalues into one where we are only interested in those eigenvalues (and corresponding eigenvectors) lying within (or in the neighborhood) of a given closed contour $\Gamma \subset \Omega$. The relevant information to approximate these eigenvalues (and corresponding eigenvectors) can be extracted from the function $T(z)$ by using (approximate) contour integrals to the resolvent operator $T(z)^{-1}$ applied to a rectangular matrix $\hat{V}$:

$$\frac{1}{2\pi i} \int_{\Gamma} f(z) T(z)^{-1} \hat{V} dz \quad \in \mathbb{C}^{m \times q}$$

for different choices of the function $f(z)$, e.g., $f(z) = z^0, z^1, z^2, \ldots$. Here, $f : \Omega \to \mathbb{C}$ is analytic in Ω and $\hat{V} \in \mathbb{C}^{m \times q}$ is a matrix chosen randomly or in another specified way, with $q \leq m$.

For a detailed overview of the history and current research on solving eigenvalue problems by contour integration, we refer the interested reader to the introduction section of [25]. Here, only some key references are mentioned without having the intention of being complete. Based on the pioneering work of Delves and Lyness [6], the author of this paper together with Peter Kravanja developed several methods to compute the zeros of a scalar analytic function $t(z)$ (for a synthesis of these results, see [14]) reducing the problem to a generalized eigenvalue problem involving a Hankel matrix as well as a shifted Hankel matrix consisting of the moments of the analytic function $t(z)$. Later on Tetsuya Sakurai joined us in our study and co-authored some papers [13, 19, 12]. To solve eigenvalue problems, Sakurai and his co-authors applied the idea of the generalized eigenvalue problem involving the Hankel and shifted Hankel matrix using moments based on the resolvent function [20, 10, 15, 9, 21, 27, 1, 2]. Eric Polizzi and co-authors also used contour integrals based on the resolvent function resulting in the FEAST algorithm [18, 24, 7].

In [25], we presented an algorithm based on contour integration to solve the corresponding eigenvalue problem. We showed that the so-called filter function plays an important role in the effectiveness of the contour integration approach. To compute the eigenvalues in the neighborhood of a branch cut, we used in [25] a filter function developed in [8] by Hale, Higham and Trefethen using detailed knowledge of complex analysis. Because this knowledge is not always readily available for someone who wants to solve a specific eigenvalue problem, we will design in this paper effective filter functions using an optimization algorithm.

Our paper is organized as follows. In Section 2 we start by summarizing the algorithm as developed in [25]. Section 3 describes the properties an effective filter function should have. In Section 4, we translate these conditions into an optimiza-

tion algorithm. Section 5 illustrates the validity of the approach by showing the results of several numerical experiments. The conclusions are given in Section 6.

2 Algorithm based on approximate contour integration

To avoid notational complexity when explaining the algorithm, we assume that the eigenvalues in which we are interested are simple. For the more general case, the interested reader is referred to Beyn's paper [5].

Given a closed contour Γ in the complex plane and a matrix-valued function $T(z)$ that is analytic inside and on Γ. We want to approximate the eigenvalues λ inside this contour and corresponding eigenvectors v of the (non)linear eigenvalue problem

$$T(\lambda)v = 0, \qquad \text{with } v \neq 0.$$

In [25], we designed a variant of Beyn's algorithm [5] showing that this eigenvalue problem can be solved via contour integration of the resolvent function $T(z)^{-1}$. Consider the following contour integrals, called moments, based on the resolvent function applied to a rectangular matrix $\hat{V}$:

$$S_p = \frac{1}{2\pi i}\int_\Gamma z^p T(z)^{-1}\hat{V}dz \quad \in \mathbb{C}^{m\times q}, \qquad p = 0, 1, \ldots, P-1 \tag{1}$$

with $\hat{V} \in \mathbb{C}^{m\times q}$ a matrix chosen randomly or in another specified way, and with $q \leq m$. Especially when m is large, to have a lower computational complexity, it is important to have a value for q such that $q \ll m$ leading to much smaller moment matrices. These contour integrals are approximated by a quadrature rule, i.e.,

$$\int_\Gamma f(z)dz \approx \sum_{j=1}^{\delta} u_j f(t_j).$$

As was explained in [5], from Keldysh' theorem, we know that the resolvent function $T(z)^{-1}$ can be written (for simple eigenvalues λ_k) as

$$T(z)^{-1} = \sum_k v_k w_k^H \frac{1}{z-\lambda_k} + R(z)$$

with $R(z)$ an analytic function where $T(z)$ is analytic, and with v_k and w_k suitably scaled right and left eigenvectors, respectively, corresponding to the (simple) eigenvalue λ_k. The case of multiple eigenvalues can be treated in a similar way but we refer to [5] for the technical details. Note that if T^{-1} is a matrix-valued strictly proper rational function, the analytic function R is equal to zero. This is the case, for example, if $T(z) = A - zB$ with B nonsingular or if $T(z)$ is a matrix polynomial in z with nonsingular highest degree coefficient.

Hence, applying the quadrature rule to approximate the moments S_p of the resolvent function $z^p T(z)^{-1}$ (and taking $\hat{V} = I$ to simplify the notation) gives us the discretized moments $\tilde{S}_p$:

$$\begin{aligned} S_p &= \int_\Gamma z^p T(z)^{-1} dz \\ &\approx \sum_{j=1}^{\delta} u_j t_j^p T(t_j)^{-1} = \tilde{S}_p \\ &= \sum_k v_k w_k^H \sum_{j=1}^{\delta} \frac{u_j t_j^p}{t_j - \lambda_k} + \sum_{j=1}^{\delta} u_j t_j^p R(t_j). \end{aligned}$$

The filter functions $b_p(z)$ are defined as the rational functions of degree δ corresponding to the quadrature rule as follows

$$b_p(z) = \sum_{j=1}^{\delta} \frac{u_j t_j^p}{t_j - z}. \tag{2}$$

To be able to easily extract the eigenvalues λ_k (and corresponding eigenvectors v_k and w_k) from the knowledge of the discretized moments $\tilde{S}_p$, $p = 0, 1, \ldots, P-1$, these filter functions $b_p(z)$ have to satisfy the following conditions:

1. $b_p(z) = b_0(z) z^p$;
2. $|\sum_{j=1}^{\delta} u_j t_j^p R(t_j)|$ is small;
3. $|b_p(z)|$ is large inside Γ and small outside Γ.

Based on this definition the discretized moments $\tilde{S}_p$ can be rewritten as

$$\begin{aligned} \tilde{S}_p &= \sum_k v_k w_k^H b_p(\lambda_k) + \sum_{j=1}^{\delta} u_j t_j^p R(t_j) && (3) \\ &= \sum_k v_k w_k^H \lambda_k^p b_0(\lambda_k) + \sum_{j=1}^{\delta} u_j t_j^p R(t_j) \qquad \text{(condition 1)} && (4) \\ &\approx \sum_k v_k w_k^H \lambda_k^p b_0(\lambda_k) \qquad \text{(condition 2).} && (5) \end{aligned}$$

Note that the first condition allows to extract the factors λ_k^p out of $b_p(\lambda_k)$ going from (3) to (4).

If the eigenvalues are ordered such that

$$|b_0(\lambda_1)| \geq |b_0(\lambda_2)| \geq \cdots \geq |b_0(\lambda_K)| > \varepsilon \geq |b_0(\lambda_{K+1})| \geq \cdots$$

and we only keep the eigenvalues for which the modulus of the filter function $b_0(z)$ is larger than a small value ε, the discretized moments $\tilde{S}_p$ can be written as

$$\tilde{S}_p \approx V \Lambda^p \hat{W}^H$$

where

$$\begin{aligned} V &= \left[\begin{array}{ccc} v_1 & \cdots & v_K \end{array}\right] \\ \Lambda &= \mathrm{diag}(\lambda_1, \ldots, \lambda_K) \\ \hat{W}^H &= \mathrm{diag}\left(b_0(\lambda_1), \ldots, b_0(\lambda_K)\right) \left[\begin{array}{ccc} w_1 & \cdots & w_K \end{array}\right]^H. \end{aligned}$$

The third condition guarantees that we keep the eigenvalues inside Γ and filter away the ones away from Γ leading to a small value for K. I.e., the remaining eigenvalues $\lambda_{K+1}, \ldots$ away from Γ (if these exist) are filtered away, i.e., $|b_0(\lambda_{K+1})|, \ldots$ is negligible. Note that in case $\hat{V}$ is different from the identity matrix, the only thing that changes is the definition of $\hat{W}$, i.e.,

$$\hat{W}^H = \mathrm{diag}\left(b_0(\lambda_1), \ldots, b_0(\lambda_K)\right) \left[\begin{array}{ccc} w_1 & \cdots & w_K \end{array}\right]^H \hat{V}.$$

Note that multiplication by $\hat{V}$ should keep the rank of $\left[\begin{array}{ccc} w_1 & \cdots & w_K \end{array}\right]^H$ unchanged.

With $\mu + \nu + 1 < P$, let

$$\begin{aligned} H &= \left[\begin{array}{cccc} \hat{S}_0 & \hat{S}_1 & \cdots & \hat{S}_\nu \\ \hat{S}_1 & & \iddots & \vdots \\ \vdots & \iddots & & \vdots \\ \hat{S}_\mu & \cdots & \cdots & \hat{S}_{\mu+\nu} \end{array}\right] \\ &\approx \left[\begin{array}{c} V \\ V\Lambda \\ \vdots \\ V\Lambda^\mu \end{array}\right] \left[\begin{array}{cccc} \hat{W}^H & \Lambda\hat{W}^H & \cdots & \Lambda^\nu\hat{W}^H \end{array}\right] \end{aligned}$$

and let

$$\begin{aligned} H^< &= \left[\begin{array}{cccc} \hat{S}_1 & \hat{S}_2 & \cdots & \hat{S}_{\nu+1} \\ \hat{S}_2 & & \iddots & \vdots \\ \vdots & \iddots & & \vdots \\ \hat{S}_{\mu+1} & \cdots & \cdots & \hat{S}_{\mu+\nu+1} \end{array}\right] \\ &\approx \left[\begin{array}{c} V \\ V\Lambda \\ \vdots \\ V\Lambda^\mu \end{array}\right] \Lambda \left[\begin{array}{cccc} \hat{W}^H & \Lambda\hat{W}^H & \cdots & \Lambda^\nu\hat{W}^H \end{array}\right] \end{aligned}$$

be defined as block Hankel matrices of block size $(\mu + 1) \times (\nu + 1)$.

Therefore, the canonical polyadic decomposition (CPD) [11] of the tensor consisting of the two slices H and $H^<$ is (approximately) given by

$$\sum_{k=1}^{K} \left[\begin{array}{c} v_k \\ v_k\lambda_k \\ \vdots \\ v_k\lambda_k^\mu \end{array}\right] \odot \left[\begin{array}{c} \overline{\hat{w}_k} \\ \lambda_k\overline{\hat{w}_k} \\ \vdots \\ \lambda_k^\nu\overline{\hat{w}_k} \end{array}\right] \odot \left[\begin{array}{c} 1 \\ \lambda_k \end{array}\right]$$

where $\hat{w}_k$ denotes the kth column vector of $\hat{W}$ for $k = 1, \ldots, K$, the notation $\overline{\cdot}$ indicates taking the complex conjugate, and $\odot$ denotes the outer product. Several other tensors can be built using the moments leading to similar CPDs. The interested reader is referred to [25]. In [5], Beyn extracted the eigenvalues and corresponding eigenvectors by reducing the problem to a linear eigenvalue problem given the matrices H and $H^<$. Using a tensor approach instead of a matrix approach can lead to a more robust algorithm [17, 16]. Tensorlab [23] provides a robust algorithm for computing the canonical polyadic decomposition. In our numerical experiments, the tensor variant has always computed solutions that are at least as accurate as via Beyn's method. However, further analysis is necessary.

The algorithm for solving eigenvalue problems via contour integration is summarized in the following steps.

1. Choose a filter function $b_0(z)$, i.e., a quadrature formula,
   depending on the domain in which the requested eigenvalues are lying and the analytic properties of the function $T(z)$.
2. Choose $\hat{V} \in \mathbb{C}^{m \times q}$ of rank q.
3. Compute the discretized moments $\tilde{S}_p$, $p = 0, \ldots, P-1$ via contour integration based on the quadrature formula.
4. Compute the canonical polyadic decomposition of a Hankel tensor constructed from the computed moments $\tilde{S}_p$.
5. The first and third factor matrix lead to the approximate eigenvectors and corresponding eigenvalues, respectively.

In [25] several numerical examples are given based on this algorithm. From these examples it becomes clear that it is important to have robust and cheap ways to design good filter functions because a user who wants to solve a nonlinear eigenvalue problem does not always have the knowledge on complex analysis to derive a suitable filter function depending on the contour Γ and the properties of the function $T(z)$. In [25] a filter function is used based on results derived in [8] using extensive knowledge of complex analysis. In the next section, we want to state the conditions that have to be satisfied by an effective filter function $b_0(z)$. In Section 4, we will translate these conditions into an optimization problem.

3 Rational filter functions

In the previous section, we argued that it is important to have efficient ways to design good filter functions without the need for the user to have an extensive knowledge of complex analysis. Repeating the previous section, this design consists in finding the weights u_j and the nodes t_j such that the following conditions are satisfied for $p = 0, 1, \ldots, P-1$:

1. $b_p(z) = b_0(z)z^p$;

2. $|\sum_{j=1}^{\delta} u_j t_j^p R(t_j)|$ is small;

3. $|b_p(z)|$ is large inside Γ and small outside Γ

with the filter functions $b_p(z)$ defined in (2).

We call the function $b_0(z)$ "the filter function" and denote it as $b(z) = \frac{n(z)}{d(z)}$ with $d(z) = \prod_j (z - t_j)$. It is clear that this filter function is uniquely determined by the parameters, the nodes t_j and the weights u_j. We show now that the three conditions above are interrelated and can be reformulated in terms of the filter function $b(z)$.

Let us look at the third condition. To get a filter function that decreases rapidly in magnitude outside the bounded domain Ω, i.e., when $|z| \to \infty$, the degree δ_n of the numerator $n(z)$ should be small compared to the degree δ of the denominator polynomial $d(z)$. The filter function $b(z)$ behaves as follows: $b(z) = \mathcal{O}(z^{\delta_n - \delta})$ when $|z| \to \infty$. Hence, the fastest decay would be when $\delta_n = 0$. Let us see how the condition $\deg n(z) = \delta_n$ is translated in terms of the parameters t_j and u_j.

$$\begin{aligned} b(z) &= \sum_{j=1}^{\delta} \frac{u_j}{t_j - z} \\ &= \sum_{j=1}^{\delta} u_j \sum_{k=1}^{\infty} t_j^{k-1} z^{-k}, \qquad |z| \to \infty \\ &= \sum_{k=1}^{\infty} \left(\sum_{j=1}^{\delta} u_j t_j^{k-1} \right) z^{-k}, \qquad |z| \to \infty \end{aligned}$$

Hence, if

$$\sum_{j=1}^{\delta} u_j t_j^{k-1} = 0, \qquad k = 1, 2, \ldots, \Delta, \tag{6}$$

then the degree δ_n of the numerator $n(z)$ satisfies $\delta_n \leq \delta - (\Delta + 1)$ and $b(z) = \mathcal{O}(z^{-(\Delta+1)})$ when $|z| \to \infty$. Conditions (6) are equivalent to the second condition above in the sense that they are equivalent to

$$\sum_{j=1}^{\delta} u_j t_j^p R(t_j) = 0,$$

for $R(z) = z^0, z^1, \ldots, z^{\Delta - p - 1}$. Let us look now at the first condition above. When conditions (6) are satisfied, we know that the degree of the numerator $n(z)$ is at least $\Delta + 1$ less than the degree δ of the denominator $d(z)$ of the filter function $b(z)$. Hence, $z^p b(z)$ is still strictly proper for $p = 0, 1, \ldots, \Delta$. Therefore, the rational function $z^p b(z)$ can be written as

$$\begin{aligned} z^p b(z) &= \sum_{l} \frac{\lim_{z \to t_j} z^p b(z)(z - t_j)}{z - t_j} \\ &= \sum_{l} \frac{u_l t_j^p}{z - t_j} \\ &= b_p(z), \qquad p = 0, 1, \ldots, \Delta \end{aligned}$$

4 Optimization algorithm

In this section, the conditions of the previous section are translated into an optimization algorithm to design an effective filter function $b(z)$. This filter function can be written as

$$b(z) = \sum_{j=1}^{\delta} \frac{u_j}{t_j - z}$$

with $u_j, t_j \in \mathbb{C}$. The conditions of the previous section can be translated in a nonlinear least squares optimization problem minimizing the following nonlinear objective function:

$$\begin{aligned} F(u,t) &= \sum_{i=1}^{N} \mu_i^2 |b(z_i) - f_i|^2 + \sum_{k=1}^{\Delta} \nu_k^2 \left| \sum_{j=1}^{\delta} u_j t_j^{k-1} \right|^2 \\ &= \sum_{i=1}^{N} |F_i(u,t)|^2 + \sum_{k=1}^{\Delta} |G_k(u,t)|^2 \end{aligned}$$

with

$$\begin{aligned} F_i(u,t) &= \mu_i (b(z_i) - f_i), \qquad i = 1, 2, \ldots, N, \\ G_k(u,t) &= \nu_k \sum_{j=1}^{\delta} u_j t_j^{k-1}, \qquad k = 1, 2, \ldots, \Delta \end{aligned}$$

with positive weights μ_i and ν_k. The first sum of the objective function is a translation in least squares terms of the condition that $|b(z)|$ is large inside Γ and small outside Γ. It is equivalent to demand that $b(z_i) \approx f_i$ for $i = 1, 2, \ldots, N$ with $0 < \mu_i \in \mathbb{R}$ the corresponding weights in the least squares approximation problem. The points z_i can be divided into two subsets. The first subset forms a discretization of the boundary of the region of interest where the modulus of the filter function $|b(z)|$ is large, i.e., approximately equal to one. This also makes the modulus of the filter function large inside the region of interest. The second subset of points z_i are chosen outside this region of interest where the modulus should be very small. Similarly, instead of covering the whole subset of the complex plane outside the region of interest, it is enough to choose a discretization of a curve around the region of interest.

The second sum in the objective function is a translation in least squares terms of conditions (6). In the next section on the numerical experiments, we will show that to obtain a good filter function, it is important to make a good choice for the weights μ_i, for the location of the points z_i, for Δ as well as for the relative size of the weights ν_k with respect to the weights μ_i. More details are given in the section on the numerical experiments.

The nonlinear least squares problem

$$\text{minimize}_{u,t} \sum_{i=1}^{N} |F_i(u,t)|^2 + \sum_{k=1}^{\Delta} |G_k(u,t)|^2$$

is solved using the Complex Optimization Toolbox [22] which is also part of Tensorlab [23]. The partial derivatives that we need, to apply the function `nls_gndl` of Tensorlab are

$$\frac{\partial F_i}{\partial u_j} = \frac{w_i}{t_j - z_i}, \qquad \frac{\partial F_i}{\partial t_j} = \frac{w_i u_j}{(t_j - z_i)^2}$$
$$\frac{\partial G_k}{\partial u_j} = t_j^{k-1}, \qquad \frac{\partial G_k}{\partial t_j} = (k-1) u_j t_j^{k-2}.$$

The optimization algorithm needs a suitable initialization where we take the nodes t_j a little bit outside Γ and the weights u_j such that the numerator of the filter function is a constant, i.e., the second sum of the objective function is zero with $\Delta = \delta - 1$, and such that the function value of the filter function is 1 for a point inside Γ. Hence, using the barycentric weights β_j, i.e., the residuals of the partial fraction description of the rational function

$$\tilde{b}(z) = \frac{1}{\prod_{j=1}^{\delta}(z - t_j)} = \sum_{j=1}^{\delta} \frac{\beta_j}{z - t_j}$$

with $\beta_j = 1/\prod_{i=1, i\neq j}^{\delta}(z - t_i)$, the weights u_j are then equal to $u_j = \alpha\beta_j$ with $\alpha = 1/\tilde{b}(\tilde{z})$ with $\tilde{z}$ the point inside Γ. E.g., when the initial points are taken such that $t_j = \rho\tilde{t}_j$ with $\tilde{t}_j$ the roots of unity, i.e., the nodes t_j are lying on a circle with center the origin and radius ρ, the corresponding weights u_j for $\tilde{z} = 0$ are $u_j = -t_j/\delta$.

5 Numerical experiments

In this section, we will use the algorithm developed in the previous section to design some effective filter functions. In the experiments, it will become clear that it is crucial to make a good choice for the weights μ_i, for the location of the points z_i, for Δ as well as for the relative size of the weights ν_k with respect to the weights μ_i.

Experiment 1: We take Γ as the unit circle. Let's look for a filter function $b(z)$ of degree $\delta = 32$. The approximation data is as follows:

z_i	f_i	μ_i
$\{$roots of $(z/r)^n - 1$ with $n = 5\delta$ and $r = 1\}$	1	1
$\{$roots of $(z/r)^n - 1$ with $n = 5\delta$ and $r = 2\}$	0	1

The parameter Δ is taken equal to $\delta - 1$ with corresponding weights $\nu_k = 1$. As initialization for the nodes t_j we take the roots of unity multiplied by $\rho = 1.1$. Figure 1 shows the behavior of the filter function in 3D and via contour lines. It turns out that the derived filter is nothing else than the well-known unit disk filter function (trapezium filter) [3] corresponding to the trapezium rule in the roots of unity but scaled with a factor which approximates $\sqrt{2}$. This is illustrated in Figure 2 where the behavior of the designed filter is compared with this unit disk filter scaled with the factor $\sqrt{2}$. Note that when the points z_i would be scaled by $1/\sqrt{2}$, we would

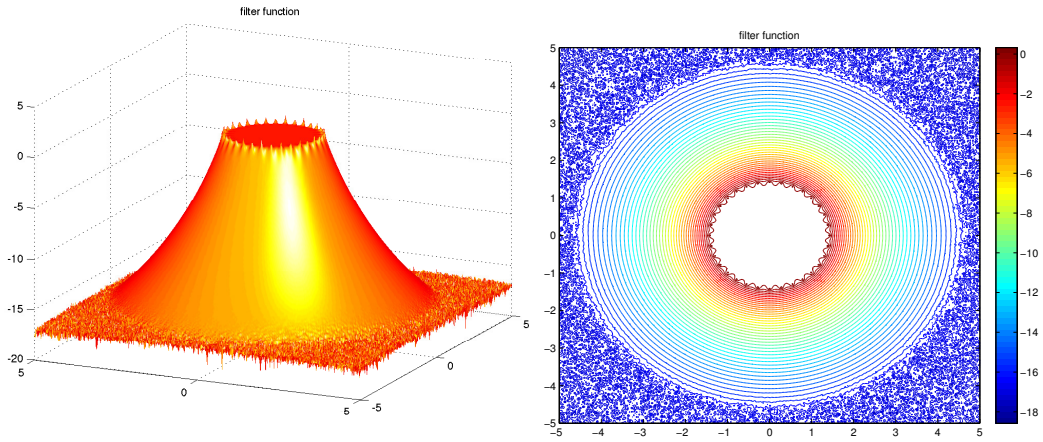

Figure 1: 3D-plot and contour lines of the log of the modulus of the computed filter function

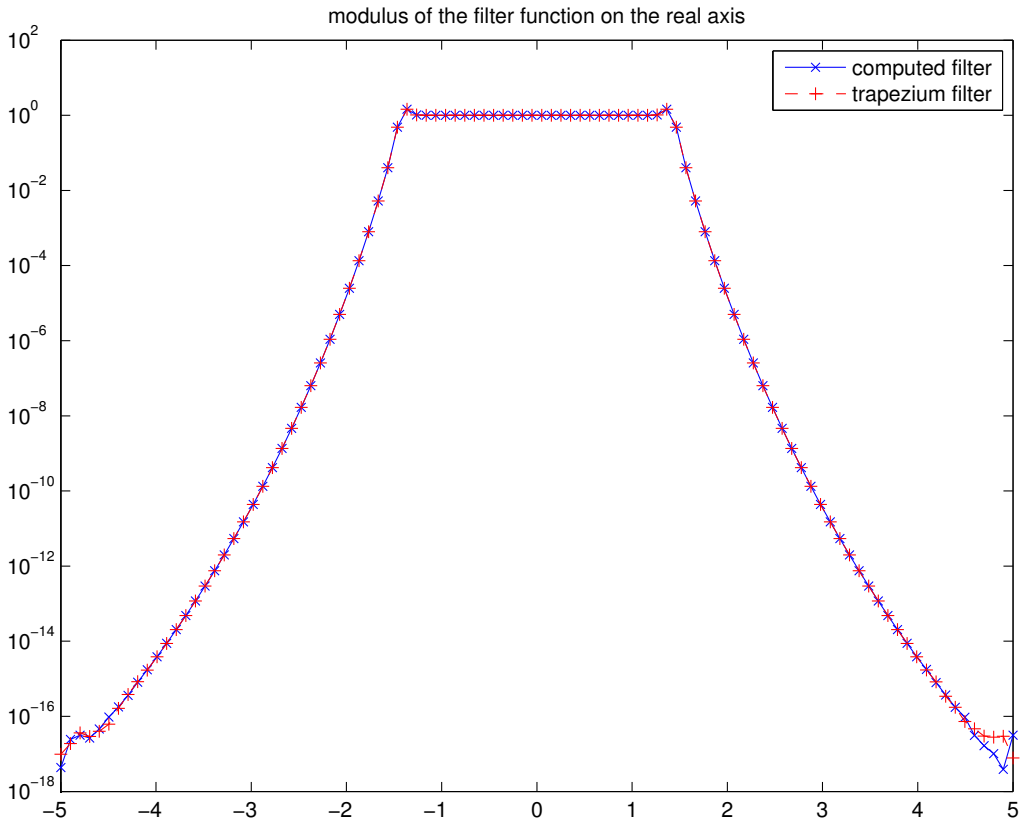

Figure 2: The behavior of the computed filter function and the classical unit disk filter on the real line

obtain a very good approximation of the unit disk filter. Another way to move the computed nodes t_j more towards the direction of Γ when solving the optimization problem is making the least squares weight μ_i connected to the points z_i having $|z_i| = 1$ smaller compared to the weight connected to the other points.

Experiment 2: In [25], we considered the gun problem from the collection of nonlinear eigenvalue problems established by Betcke, Higham, Mehrmann, Schroeder and Tisseur [4]. This problem is related to a model of a radio-frequency gun cavity. The problem size is equal to $m = 9956$ whereas the function T has the following form:

$$T(z) = \begin{bmatrix} K & M & iW_1 & iW_2 \end{bmatrix} \begin{bmatrix} 1 \\ -z \\ \sqrt{z} \\ \sqrt{z-\alpha} \end{bmatrix}$$

where K, M, W_1 and W_2 are sparse matrices, and $\alpha = (108.8774)^2$.

We would like to approximate all the eigenvalues located inside and in the neighborhood of the circle that is symmetric with respect to the real axis, and that intersects the real axis at $\alpha = (108.8774)^2$ and $\beta = 340^2$. Because α is a branch point of the function $T(z)$, we want to design a filter whose modulus is of the size of the machine precision in α (and for real values smaller than α) and whose magnitude increases as fast as possible when going to the right of the point α. The filter function should have a magnitude of order 1 inside the circle considered between the points α and β. Using a linear transformation, we can map the original problem as a problem where we choose Γ as the unit circle and where the point α maps to a point on the real line less than -1 but as close to it as possible and where β is mapped on a real value larger than 1. We design a filter function $b(z)$ with degree $\delta = 32$. After some trial and error, it turns out that an effective filter function is obtained in two optimization steps. In the first step the parameter Δ is taken equal to zero while the other parameters are chosen as

z_i	f_i	w_i
$\{$roots of $(z/r)^n - 1$ with $n = 4\delta$ and $r = 1\} \setminus \{1\}$	1	10^{-10}
$\{\ 1\ \}$	1	10^{-1}
$\{$roots of $(z/r)^n - 1$ with $n = 4\delta$ and $r = 4\}$	0	1
$\{-4, \ldots, 1.2\}$ (4δ equidistant numbers)	0	1

As initialization for the nodes t_j we take the roots of unity multiplied by $\rho = 1.1$. In the second optimization step the solution obtained in the first step is taken as initialization with the same parameters as before except for the parameter Δ which is taken equal to $\delta/2 = 16$ with corresponding weights $\nu_k = 1$. Figure 3 shows the behavior of the computed filter function in 3D and via contour lines. This can be compared with a similar plot in Figure 4 for the (scaled and shifted) Hale-Higham-Trefethen filter (HHT-filter) as developed in [8] and used in [25] as mentioned before. The behavior on the real line is illustrated in Figure 5 comparing the designed filter with the classical unit disk filter and the HHT-filter. Looking at these figures, it is clear that for a comparable size of the domain in which the magnitude of the filter function is of order 1, the computed filter has a magnitude of the order of

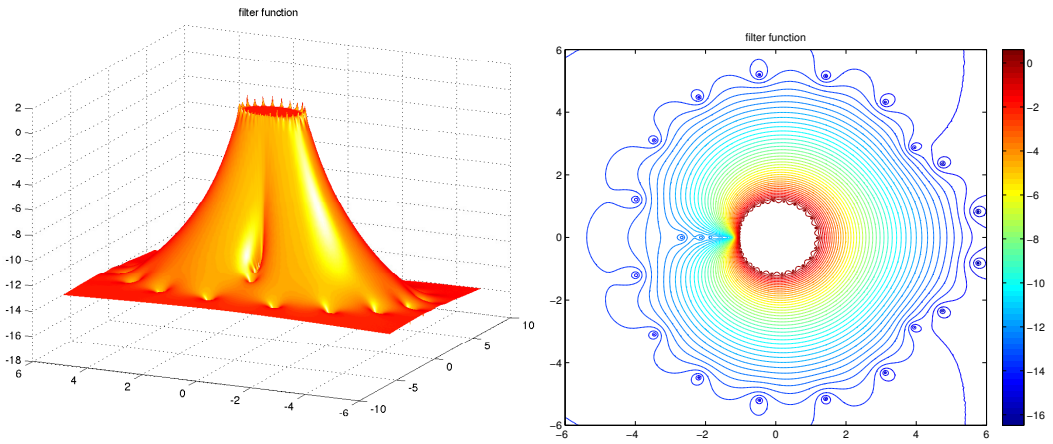

Figure 3: 3D-plot and contour lines of the log of the modulus of the computed filter function

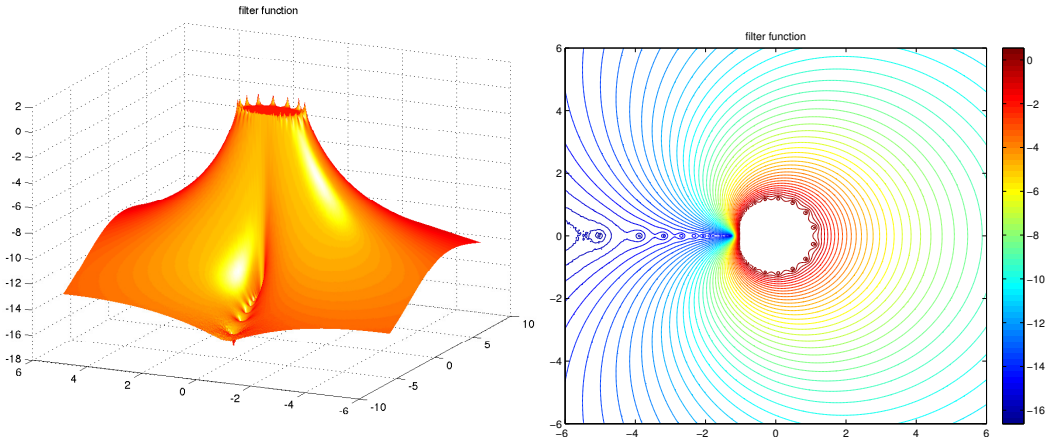

Figure 4: 3D-plot and contour lines of the log of the modulus of the Hale-Higham-Trefethen filter function

the machine precision outside a circle with center the origin and radius 5 while the HHT-filter has a much wider influence region. On the other hand, the HHT-filter has a magnitude of the order of the machine precision on the branch cut while the computed filter is three orders of magnitude larger.

We will compare now the effectiveness of using the computed filter function and the HHT-filter to solve the gun problem by the algorithm developed in Section 2. As in [25], only two discretized moments $\tilde{S}_p$, $p = 0, 1$ are computed where q is taken equal to 40. Hence, $q = 40$ (or less) eigenvalues can be approximated. To compare the accuracy of these computed eigenvalues, the relative residual is considered. This relative residual for each of the computed eigenvalues $\tilde{\lambda}_k$ with corresponding eigenvector $\tilde{v}_k$ is given by

$$\epsilon_k = \frac{\|T(\tilde{\lambda}_k)\tilde{v}_k\|_1}{\|T(\tilde{\lambda}_k)\|_1\|\tilde{v}_k\|_1}.$$

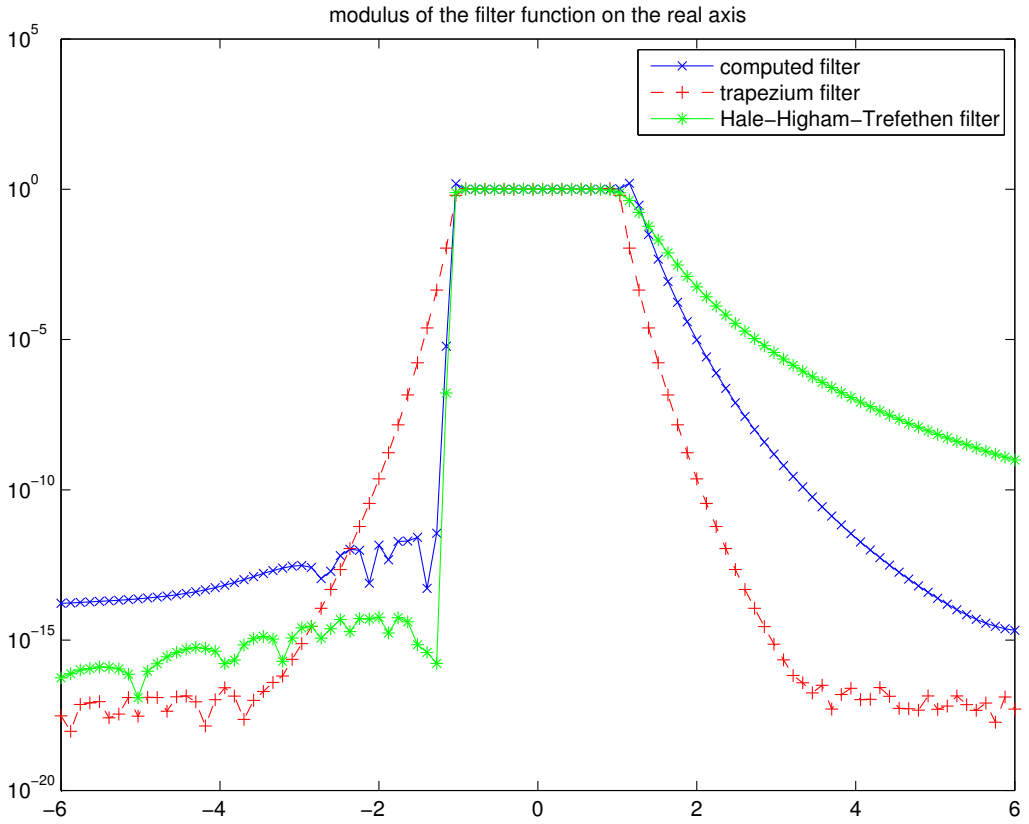

Figure 5: The behavior of the computed filter function, the classical unit disk filter and the Hale-Higham-Trefethen filter on the real line

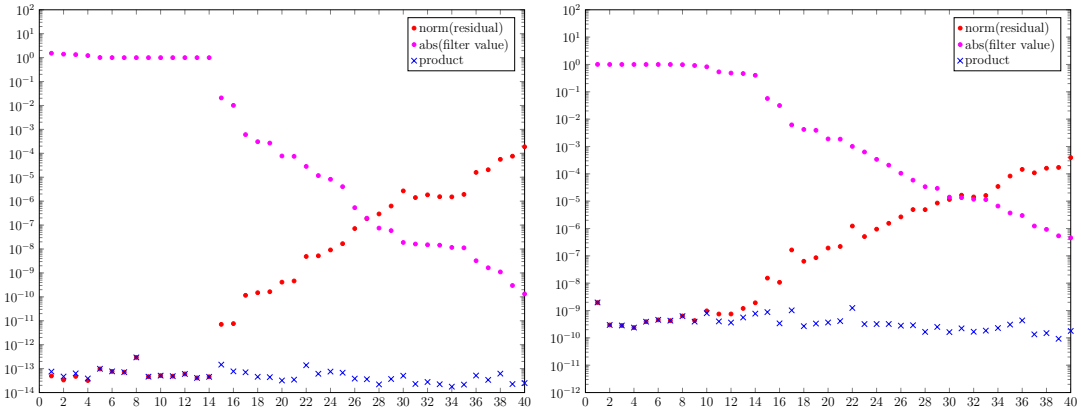

Figure 6: Relative residuals for the computed eigenvalues for the computed (left) and HHT-filter (right)

Figure 6 plots these relative residuals as well as the magnitude of the filter function in each of these eigenvalues, and their corresponding product. Because the accuracy of $\tilde{\lambda}_k$ depends on the extraction of the corresponding information from $\tilde{S}_0$ and $\tilde{S}_1$, this accuracy is limited to $|b(\tilde{\lambda}_k)|$. As was argued in [25], we expect that the product of the relative residual ϵ_k and the magnitude of the corresponding filter function $|b(\tilde{\lambda}_k)|$ behaves as a constant, i.e.,

$$\epsilon_k |b(\tilde{\lambda}_k)| \approx C_b$$

with C_b a constant defined on the filter function used. Because the computed filter filters away the eigenvalues four orders of magnitude better compared to the HHT-filter as can be seen from the magnitude of the filter function for $\tilde{\lambda}_{40}$, the constant C_b will be four orders of magnitude smaller for the computed filter compared to the HHT-filter.

Figure 7 shows the computed eigenvalues using the computed and HHT-filter on the contour plot of the corresponding filter. The two filled red dots indicate the two points corresponding to α and β of the gun problem. The blue circles represent the computed eigenvalues $\tilde{\lambda}_k$.

Experiment 3: In [26], the problem is considered of computing all characteristic roots of a delay differential equation (DDE) in a given right half plane. In this experiment we consider Example 7 of this paper, where all eigenvalues are looked for having their real part larger than -1 for an analytic function $T(z)$ of size 20×20 having 9 delays.

Let us first construct an effective unit square instead of the unit disk filter function with degree $\delta = 32$. This filter will be shifted and scaled to use for the specific DDE-problem.

After some trial and error, it turns out that an effective filter function is obtained by choosing the following parameters in the optimization process

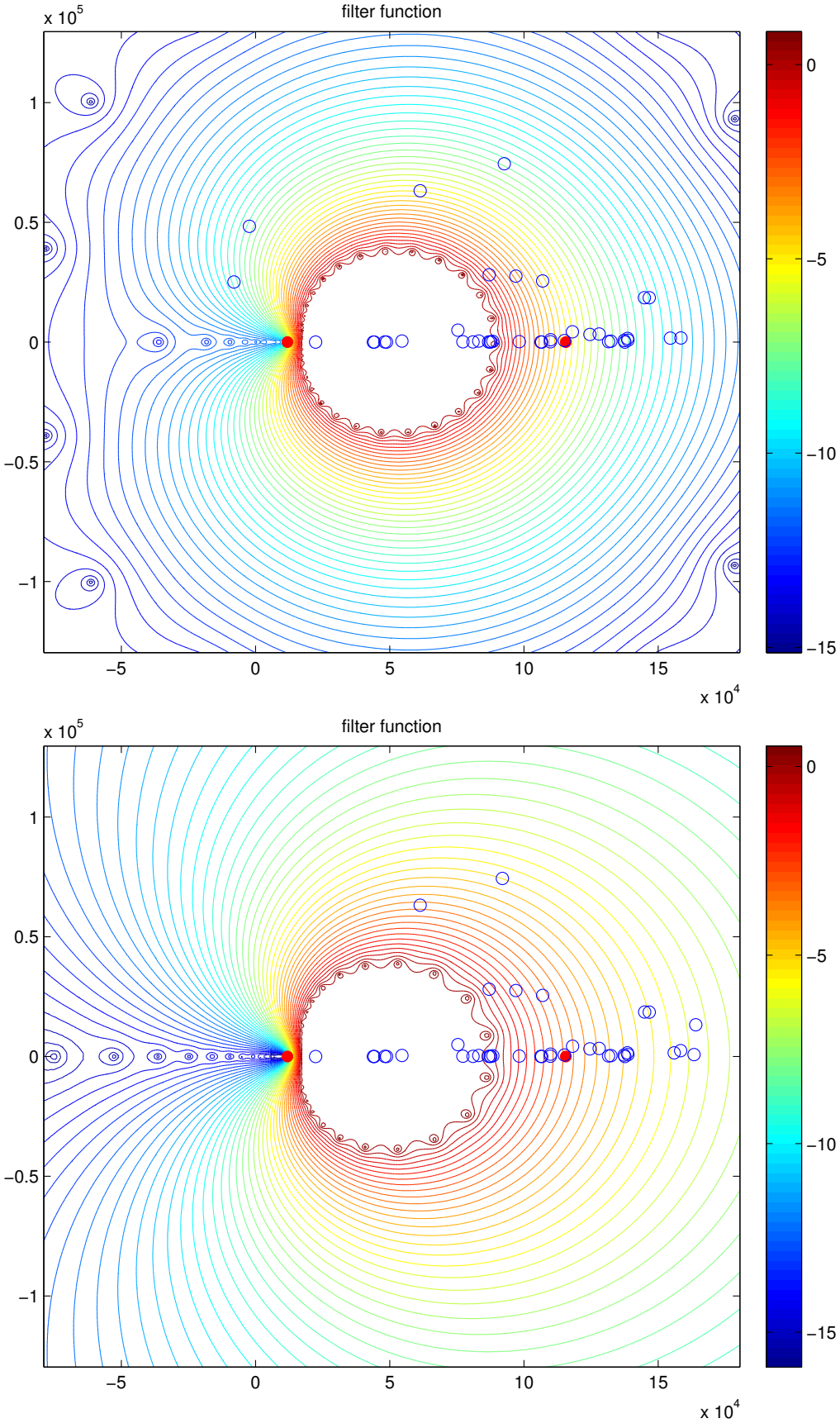

Figure 7: Computed eigenvalues using the computed (top) and HHT-filter (bottom)

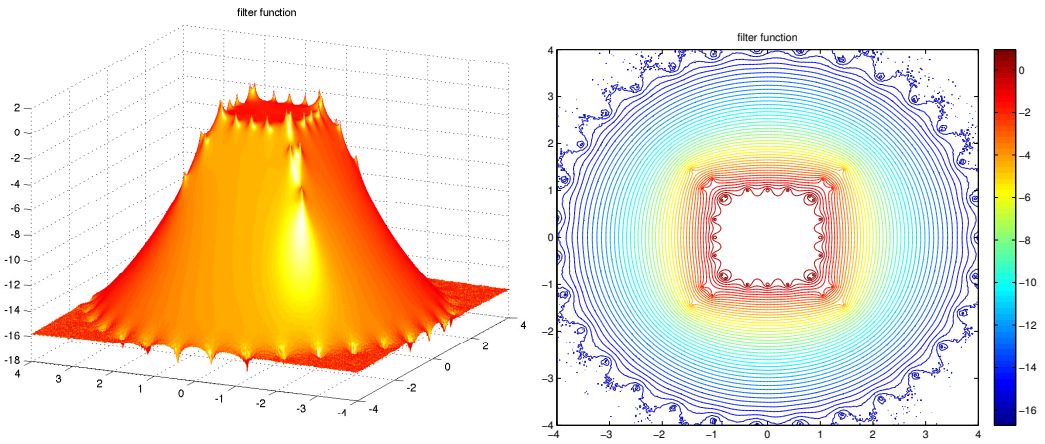

Figure 8: 3D-plot and contour lines of the log of the modulus of the computed filter function

z_i	f_i	w_i
$\{4\delta$ equidistant points on the square with side-length 2$\}$	1	10^{-10}
$\{4\delta$ equidistant points on the square with side-length 4$\}$	0	1
$\{0\}$	1	10^{-7}

As initialization for the nodes t_j we take $\delta = 32$ equidistant points on the square with each side of length 3.0. As before, in the first step of the optimization procedure Δ is taken equal to 0. Then, the obtained solution is used as initialization for the second optimization step with $\Delta = \delta - 1 = 31$.

Figure 8 shows the behavior of the computed filter function in 3D and via contour lines. The behavior on the real line is illustrated in Figure 9 comparing the designed filter with the classical unit disk filter.

We will use the computed filter now to approximate the eigenvalues with real part greater than -1.0. To this end, the square filter is shifted and scaled such that the points $\alpha = -1.0 + 7.5i$ and $\beta = 10.0 + 7.5i$ from the problem space are mapped to the point -1.0 and 0.75 of the filter space, respectively. The idea is to approximate only the requested eigenvalues in the upper half plane because the eigenvalues occur as real eigenvalues or in conjugate pairs for this specific DDE-problem. Because this maps a lot of eigenvalues in the neighborhood of the unit square, we need to take more than two moments because the size of the matrices involved is only 20×20. Taking 18 moments, the relative residuals as well as the magnitude of the filter function in each of these eigenvalues, and their corresponding product is shown in Figure 10.

Figure 11 shows the computed eigenvalues with relative residual smaller than 10^{-2} using the computed filter on the contour plot of the computed filter. The two filled red dots indicate the two points corresponding to α and β of the DDE-problem. The blue circles represent the computed eigenvalues $\tilde{\lambda}_k$. To be able to compare the computed eigenvalues with $\Re(z) \geq -1$ with the ones shown in Figure 9 of [26], we plot our computed eigenvalues again in Figure 12 but with a different scale for the real and imaginary axis. Note that our procedure computes all eigenvalues in the

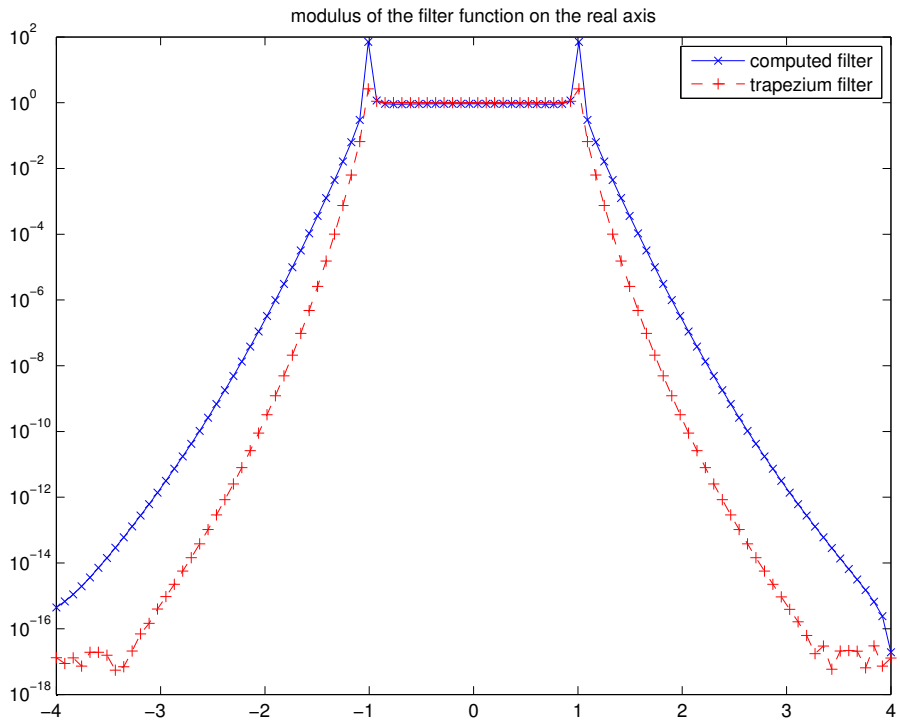

Figure 9: The behavior of the computed filter function and the classical unit disk filter on the real line

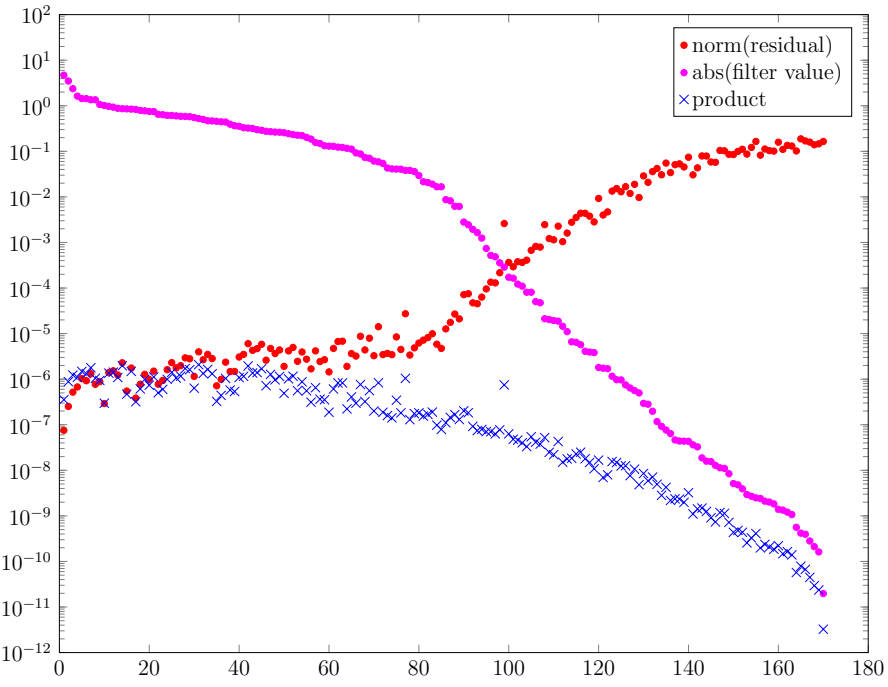

Figure 10: Relative residuals for the computed eigenvalues for the computed filter

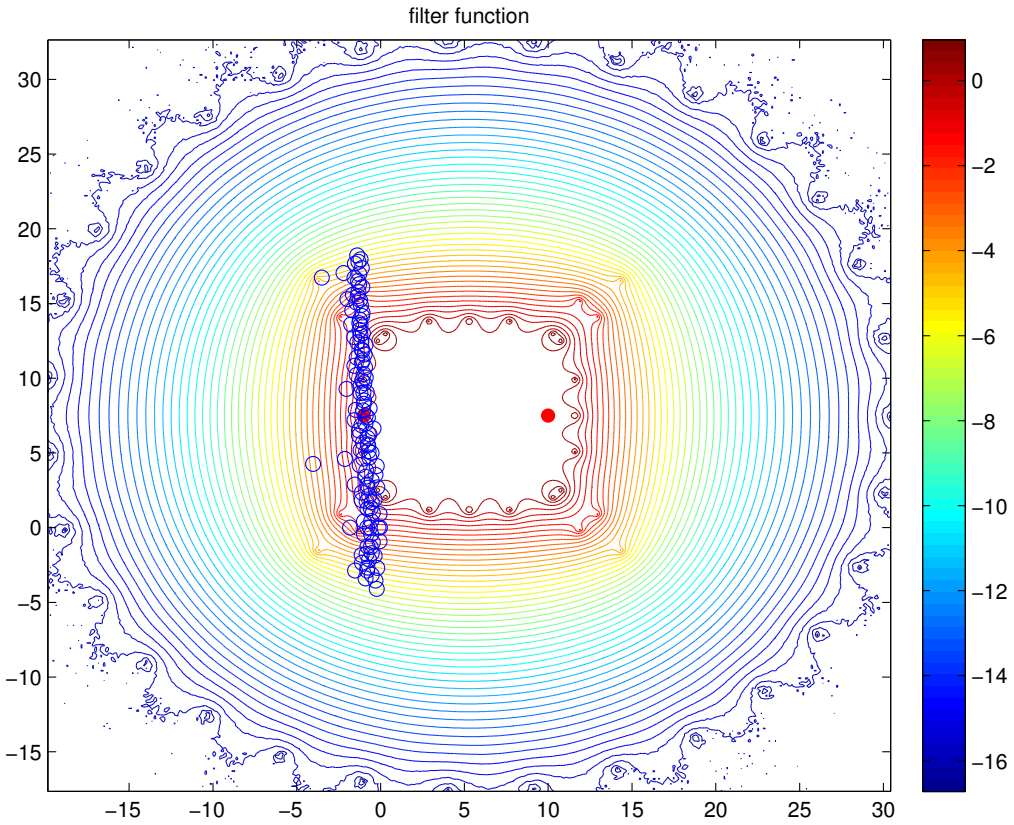

Figure 11: Computed eigenvalues using the computed filter

considered region as can be seen by comparing the two figures.

6 Conclusions

The main aim of this paper was to indicate that good rational filter functions can be computed using (nonlinear least squares) optimization techniques as opposed to designing those functions based on a thorough understanding of complex analysis. The conditions that such an effective filter function should satisfy, were derived and translated in a nonlinear least squares optimization problem solved by optimization algorithms from Tensorlab. Numerical experiments illustrated the validity of this approach. However, it is still difficult for a non experienced user to effectively choose the parameters of the optimization problem, especially the relative values of the least squares weights. This is a topic for future research.

References

[1] J. Asakura, T. Sakurai, H. Tadano, T. Ikegami, and K. Kimura. A numerical method for nonlinear eigenvalue problems using contour integrals. *JSIAM Letters*, 1:52–55, 2009.

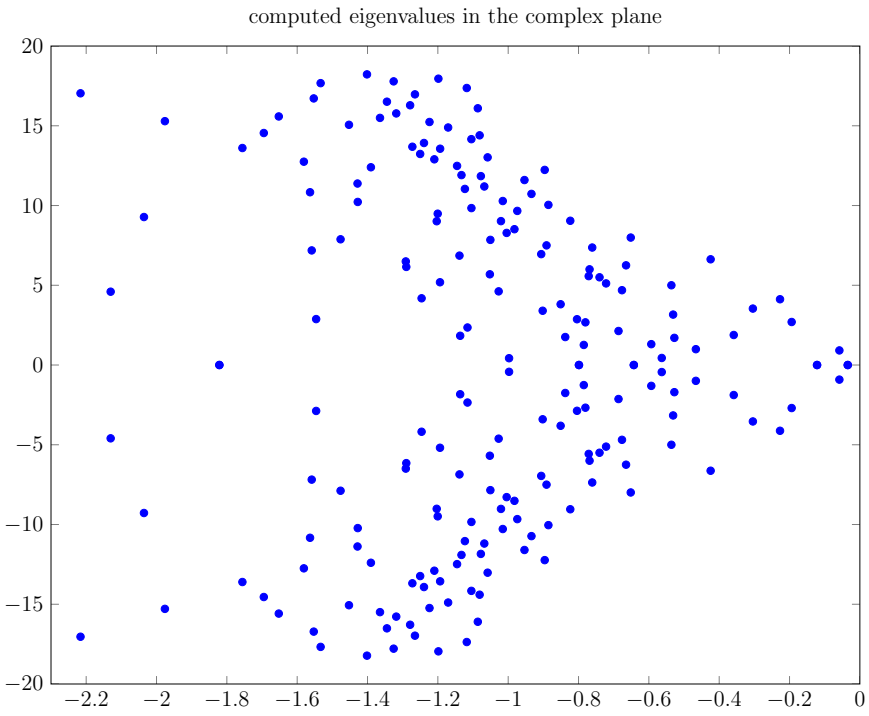

Figure 12: Computed eigenvalues using the computed filter

[2] J. Asakura, T. Sakurai, H. Tadano, T. Ikegami, and K. Kimura. A numerical method for polynomial eigenvalue problems using contour integrals. *Japan J. Indust. Appl. Math.*, 27:73–90, 2010.

[3] A. P. Austin, P. Kravanja, and L. N. Trefethen. Numerical algorithms based on analytic function values at roots of unity. *SIAM Journal on Numerical Analysis*, 52(2):1795–1821, 2014.

[4] T. Betcke, N. J. Higham, V. Mehrmann, Chr. Schröder, and F. Tisseur. NLEVP: A collection of nonlinear eigenvalue problems. MIMS EPrint 2011.116, Manchester Institute for Mathematical Sciences, The University of Manchester, December 2011.

[5] W.-J. Beyn. An integral method for solving nonlinear eigenvalue problems. *Linear Algebra and its Applications*, 436:3839–3863, 2012.

[6] L. M. Delves and J. N. Lyness. A numerical method for locating the zeros of an analytic function. *Mathematics of Computation*, 21:543–560, 1967.

[7] S. Güttel, E. Polizzi, P. T. P. Tang, and G. Viaud. Zolotarev quadrature rules and load balancing for the FEAST eigensolver. MIMS EPrint 2014.39, Manchester Institure for Mathematical Sciences, The University of Manchester, July 2014.

[8] N. Hale, N. J. Higham, and L. N. Trefethen. Computing A^α, $\log(A)$, and related matrix functions by contour integrals. *SIAM Journal on Numerical Analysis*, 46(5):2505–2523, 2008.

[9] T. Ikegami and T. Sakurai. Contour integral eigensolver for non-Hermitian systems: A Rayleigh-Ritz-type approach. *Taiwanese J. Math.*, 14(3A):825–837, 2010.

[10] T. Ikegami, T. Sakurai, and U. Nagashima. A filter diagonalization for generalized eigenvalue problems based on the Sakurai-Sugiura projection method. *Journal of Computational and Applied Mathematics*, 233:1927–1936, 2010.

[11] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September 2009.

[12] P. Kravanja, T. Sakurai, H. Sugiura, and M. Van Barel. A perturbation result for generalized eigenvalue problems and its application to error estimation in a quadrature method for computing zeros of analytic functions. *Journal of Computational and Applied Mathematics*, 161:339–347, 2003.

[13] P. Kravanja, T. Sakurai, and M. Van Barel. On locating clusters of zeros of analytic functions. *BIT*, 39(4):646–682, December 1999.

[14] P. Kravanja and M. Van Barel. *Computing the zeros of analytic functions*, volume 1727 of *Lecture Notes in Mathematics*. Springer, 2000.

[15] H. Ohno, Y. Kuramashi, T. Sakurai, and H. Tadano. A quadrature-based eigensolver with a Krylov subspace method for shifted linear systems for Hermitian eigenproblems in lattice QCD. *JSIAM Letters*, 2:115–118, 2010.

[16] J. M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential data fitting using multilinear algebra: the single-channel and the multi-channel case. *Numerical Linear Algebra with Applications*, 12(8):809–826, October 2005.

[17] J. M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential data fitting using multilinear algebra: the decimative case. *Journal of Chemometrics*, 23(7-8):341–351, 2009.

[18] E. Polizzi. Density-matrix-based algorithm for solving eigenvalue problems. *Physics Review B*, 79, 2009.

[19] T. Sakurai, P. Kravanja, H. Sugiura, and M. Van Barel. An error analysis of two related quadrature methods for computing zeros of analytic functions. *Journal of Computational and Applied Mathematics*, 152:467–480, 2003.

[20] T. Sakurai and H. Sugiura. A projection method for generalized eigenvalue problems using numerical integration. *Journal of Computational and Applied Mathematics*, 159:119–128, 2003.

[21] T. Sakurai and H. Tadano. CIRR: A Rayleigh-Ritz type method with contour integral for generalized eigenvalue problems. *Hokkaido Math. J.*, 36:745–757, 2007.

[22] L. Sorber, M. Van Barel, and L. De Lathauwer. Complex optimization toolbox, version 1.03, May 2014.

[23] L. Sorber, M. Van Barel, and L. De Lathauwer. Tensorlab v2.0, January 2014.

[24] P. T. P. Tang and E. Polizzi. FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection. to appear in SIAM J. Matrix Anal. Appl.

[25] M. Van Barel and P. Kravanja. Nonlinear eigenvalue problems and contour integrals. Technical Report TW656, Department of Computer Science, KU Leuven, October 2014.

[26] Z. Wu and W. Michiels. Reliably computing all characteristic roots of delay differential equations in a given right half plane. *Journal of Computational and Applied Mathematics*, 236:2499–2514, 2012.

[27] S. Yokota and T. Sakurai. A projection method for nonlinear eigenvalue problems using contour integrals. *JSIAM Letters*, 5:41–44, 2013.